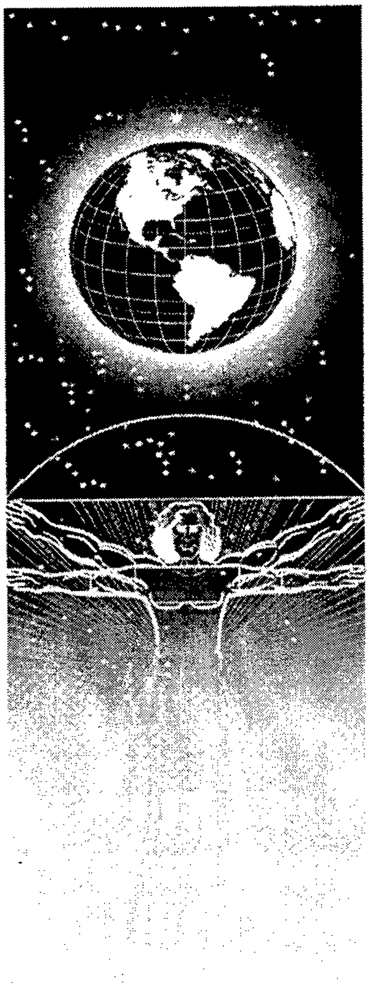
AL/HR-TR-1996-0139



UNITED STATES AIR FORCE
ARMSTRONG LABORATORY

EAGLE VIEW

Patrick K. Clark

TASC, Inc.
2555 University Blvd.
Fairborn, OH 45324

Todd Carrico
John DiPasquale

HUMAN RESOURCES DIRECTORATE
LOGISTICS RESEARCH DIVISION
2698 G Street
Wright-Patterson AFB OH 45433-7604

August 1997

Final Report for the Period February 1996 to November 1996

20010427 054

Approved for public release; distribution is unlimited.

Human Resources Directorate
Logistics Research Division
2698 G Street
Wright-Patterson AFB OH 45433-7604

NOTICES

TECHNICAL REVIEW AND APPROVAL

AL/HR-TR-1996-0139

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

ALBERT S. TORIGIAN, Lt Col, USAF
Deputy Chief
Deployment and Sustainment Division
Air Force Research Laboratory

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1997	Final - February 1996 - November 1996

4. TITLE AND SUBTITLE
Eagle View

5. FUNDING NUMBERS
C: F33615-92-D-1052
PE: 62202F
PR : 1710
TA: D2
WU: 08

6. AUTHOR(S)
Patrick K. Clark, Todd Carrico, John DiPasquale

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
TASC, Inc.
2555 University Blvd.
Fairborn, OH 45324

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Armstrong Laboratory
Human Resources Directorate
Logistics Research Division
2698 G Street
Wright-Patterson AFB, OH 45433-7604

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AL/HR-TR-1996-0139

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The objectives of the Eagle View research were to develop and test new user interfaces concepts for integrated logistics information within the Wing Operations Center (WOC), investigate the use of maintenance and logistics objects to represent the current status and capability of the wing logistics system, and provide this information to decision makers in a more easily used format. Information retrieval and presentation concepts were developed. The Integrated Model Development Environment was then used to store, call, control, and manipulate the various objects required to support and provide the required information. The Eagle View effort successfully demonstrated the capabilities and potential of an interactive wing level logistics management and planning tool.

14. SUBJECT TERMS
Logistics Command and Control — IMDE
Integrated Model Development Environment — User Interface

15. NUMBER OF PAGES
20

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Standard Form 298 (Rev. 2-89) (EG)
Prescribed by ANSI Std. 239.18
Designed using Perform Pro, WHS/DIOR, Oct 94

THIS PAGE LEFT INTENTIONALLY BLANK

Preface

This report documents the results of a study to develop and test new user interface concepts for integrated logistics information within the Wing Operations Center and the application of object based modeling technology to store, call, control, and manipulate logistics information for use by senior logistics managers. The research was conducted by TASC, Inc. under contract F33615-92-D-1052. Captain Todd Carrico, AL/HRGO was the task manager.

Table Of Contents

List Of Figures

THIS PAGE LEFT INTENTIONALLY BLANK

1. Background

As air bases become more automated, an infrastructure is coming into existence that allows access to the whole spectrum of information used by a wing commander and his or her staff. Current methods of obtaining that information involve accessing several different systems to get a complete view of the state of the wing. These systems include logistics databases and status information. The goal of this task was to show how real data could be integrated into a single application to give planners and logisticians an accurate view of the processes related to aircraft operations. The view would be in real time, as if the planners were hovering over the base and could zoom in on any area of interest, like an aerial observer - hence the name Eagle View.

2. Introduction

The purpose of Eagle View is to develop a prototype system that could demonstrate the concept of an interactive wing logistics planning tool. This tool will allow a group of wing-level logisticians to graphically construct a base layout for reception planning; set up resource levels for base supplies, equipment, and personnel; update resource levels in real-time with real data; and evaluate how well the base structure will execute a specified set of plans.

3. Eagle View Architecture

3.1. Overall Architecture

Many capabilities are integrated into the overall Eagle View architecture. Some of these capabilities are separated into different processes. The architecture provides several advantages. The different processes may run on separate CPU's, either on different workstations or on a multi-processor workstation, thus allowing the processes to run more efficiently. The architecture also creates a greater level of modularity, which allows the different capabilities of Eagle View to be developed independently. This modularity also allows the individual capabilities to be upgraded independently. As shown in Figure 1., the main components of Eagle View are:

- Real-time Simulation
- Assessment Simulation
- Simulated Data Injector
- Expert System
- Display Engine

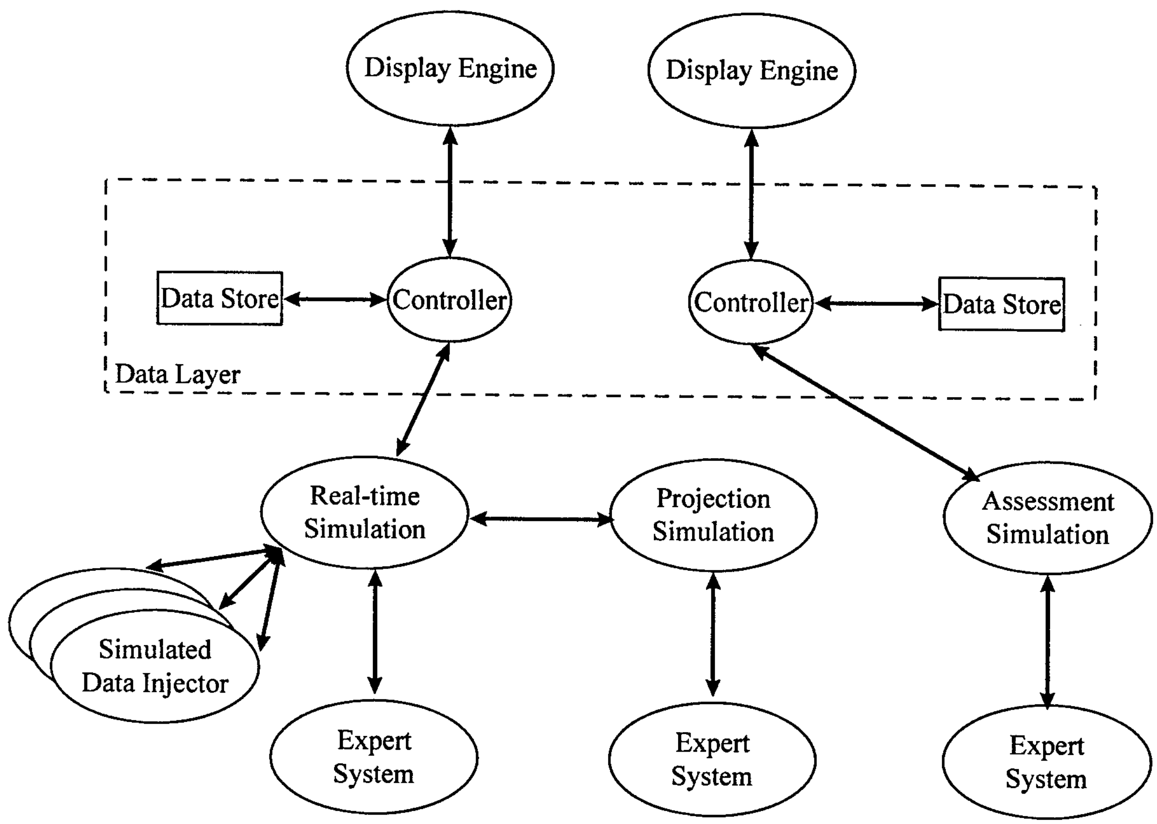


Figure 1. Eagle View Architecture

The architecture diagram shows the logical structure of the Eagle View system. The implemented architecture includes five separate components: the real-time simulation, the assessment simulation, plans, the simulated data injector, and the expert system. The controller functionality is split between the display engine and the simulation, with the data store residing in the display engine.

3.2. *Real-time Simulation*

This component of the Eagle View architecture houses the simulation engine that models the overall activity on the base. The real-time simulation provides information on the nature of the various simulation objects observed by Eagle View (for example, aircraft, hospitals, munitions storage) and the state attributes possessed by those objects (flying/idle/in maintenance state for the aircraft, supply level for the hospital, etc.).

There are three ways that these states can change during the simulation. First, in the absence of external events, the simulation possesses the logic to interpolate certain values and change state variables over time if needed. This simulation is an object-oriented discrete event simulation that was created using IMDE.

Second, the real-time simulation has the capability to receive data from external sources. The data receivers take in information while the simulation is running. The information is processed immediately so there is a dead-reckoning between the simulation and the real world. The information is received in a generic format that can be translated into an action in the simulation. The actions call a member function of a simulation object. For purposes of the prototype demonstration, there is a fixed set of events that can be translated by the data receivers. For the Eagle View project, the data receivers receive data from the simulated data injector. In a production system, the simulated data injectors will be replaced by real data feeds. The format of the data sent to the data receivers will remain the same.

A third way to direct the simulation and its states is with the use of plans. A specific plan entered into the simulation provides information on known future events and how to handle them. One example of a plan would be an aircraft flying schedule; another example is a supply ordering schedule for a hospital. Plans provide the capability to change or add to the logic in the simulation without re-compiling the simulation.

The real-time simulation connects and communicates with the other parts of the Eagle View system through the data layer, as shown in Figure 1. When the state of an object is changed, that change would be communicated to the data layer. This would allow the data layer to notify the display engines that a display needs to be updated.

Another feature of the real-time simulation is a "warp" mechanism, the capability to speed up to a certain point in the simulation. Normally, the real-time simulation clock will mirror a real clock to monitor all events and provide up-to-date status information. However, for demonstration purposes, it may be desirable at times to run the simulation faster than real time.

3.3. Assessment Simulation

The assessment simulation component of Eagle View provides the user with the capability to evaluate how a current plan, or a new, altered plan, will affect the simulation in the long-term future. The assessment simulation can be run concurrently with the real-time simulation.

The assessment simulation tracks the same state values as the real-time simulator, but at a faster speed. In a production system, multiple replications of the assessment would be performed to create confidence intervals for the future values of the state variable. In this prototype system, Eagle View allows a single run of the assessment simulation as fast as possible, and creates a separate base map.

The two initial main drivers to the assessment simulation are the plans and the current state values of the real-time simulation. At the beginning of the assessment, the user chooses a plan to assess. The plan can be associated with a single, specific object, or may specify actions and/or plans for other objects as well. In the assessment, plans cannot be changed while the simulation is in progress (unlike the real-time simulation). When the replications of the assessment are complete, the user is able to view statistics of the

attributes in the model. After the assessment, the user can substitute the real-time simulation plans for the plans used in the assessment.

As in the real-time simulation, the assessment simulation processes injected events. The events injected into the assessment simulation will come from the simulated data injector, rather than real-world data feeds. Eventually, the data files processed by the data injector will come from historical data or from data modeling a specific scenario to be tested.

3.4. Plans

The assessment capabilities of Eagle View are very important in giving users a "heads-up" as to what their situation will look like weeks or months in the future, starting with the current real situation. The user must have the ability to perform "what-if" assessments starting from the current real state; this is provided through the capability to create and edit plans for different entities (vehicles, functional organizations, shops, equipment, etc.). The current active plan, along with the injected events, drives the real-time simulation.

During the course of the real-time simulation execution, the expert system component evaluates the current system state vs. threshold values and triggers rules that determine whether to notify the user of a developing situation that requires attention. The user can modify existing plans to reflect actions that will alleviate the problem condition, or possibly reflect a new mission execution directed by wing staff. The modified plans are input into the assessment simulation along with the current base state. The assessment simulation then provides statistical output on the effect of the new plan on resource levels, mission accomplishment, etc.

New or modified plans may also be generated to improve existing processes, even in the absence of problem situations. The user can analyze the assessment output to determine whether the modified plan "works" for its intended purpose. If it does, the user can then

put the modified plan into execution by substituting it for the existing plan currently feeding the real-time simulation. Objects in the simulation have a standard member function which can be called upon to load and execute the new plan.

The user can select entities on the primary display to modify and assess either the currently-executing or alternate plans in the plan editor. Plans are developed using the Interpreted Plan Language developed by Battelle. This language allows variable assignments, asynchronous event taskings, time passage, and normal process control structures (if-then, loops, etc.). Each plan-capable object is descended from the parent-controllable object and given a unique object identification within the real-time and assessment simulations, which allows the user to pinpoint which object's plan to modify. A graphical plan creation/editing tool developed under separate contractual effort has been incorporated to provide graphical plan editing capability.

3.5. Simulated Data Injector

The simulated data injector is used to simulate the effect of real-time data feeds. The simulated data is an independent simulation object. The data injector injects events into the real-time simulation as if those events had actually happened. It is synchronized with the execution speed of the real-time simulation. Multiple data injectors can be used to simulate multiple real data feeds.

The simulated data injector parses an ASCII file which contains events defining a test scenario on which to run the Eagle View prototype demonstration. The events are of several types. One type of event is a change in the level of a resource. Other more generic events may cause a method of some object in the database to be called. For example, an event representing the crash of an aircraft may cause the simulation to call the crash() member function of an aircraft object. Each event includes a time at which the event is to occur. Using the dictated time, the data injector schedules an activity for each event listed in the file. When an activity is activated by the simulation engine, the activity sends the information associated with the event to the data receiver in the real-time simulation. This

appears to the real-time simulation as if a real world event has just occurred that the simulation must process. The real-time simulation is unaware of whether the event was initiated by a real data feed or the simulated data injector. In a production system's real-time simulation, the simulated data injector will eventually be replaced by processes which monitor real data feeds. The information obtained from real data feeds will then be translated into an event that can be passed to the real-time simulation.

3.6. Expert System

An expert system connected to both the real-time and assessment simulation components of the Eagle View system has been implemented. The purpose of the expert system is to automatically track key resource levels to determine when certain thresholds have been breached. The user will then be immediately notified when breaches occur and be allowed to make any necessary plan changes to minimize resource constraints during the simulated activity.

The CLIPS system and the work already done by Battelle have been used as starting points for the development of this expert system. Rules have been developed to model the thresholds appropriate to the simulation setting. In addition, rules can be developed to handle both resource shortage and surplus. These rules would not only alert the user to certain situations that require immediate attention, but also could point the user in the directions necessary to solve the problem. For example, the user could be notified when the expert system detects that a food shortage is developing when the ration levels drop below 7000 units and supply planes are only replenishing at a rate of 400 units per day. The user then needs to determine how to get more rations on the supply planes. Meanwhile, another threshold may be breached when the number of spare F-15 tires surpasses thirty and seven additional tires are arriving every day, thereby creating a surplus. The user may be able to modify plans to increase the amount of food arriving every day by decreasing the number of spare tires carried on each supply plane.

Figure 2 shows how the expert system integrates the classes developed for the two simulations. Simulation classes that need to interact with the expert system are assigned a CLIPS_Client attribute. The client is told which attributes of the class will be "monitored," or examined by the expert system. The class also contains a method that is registered with the client, which allows the client to execute the method when any of the monitored attributes change. The method then sends messages to a CLIPS server process via inter-process communication (IPC) methodologies such as pipes or sockets. The CLIPS server then runs the data through the expert system and returns the results to the client. This allows any changes in the state of the simulation classes to be immediately processed by the expert system.

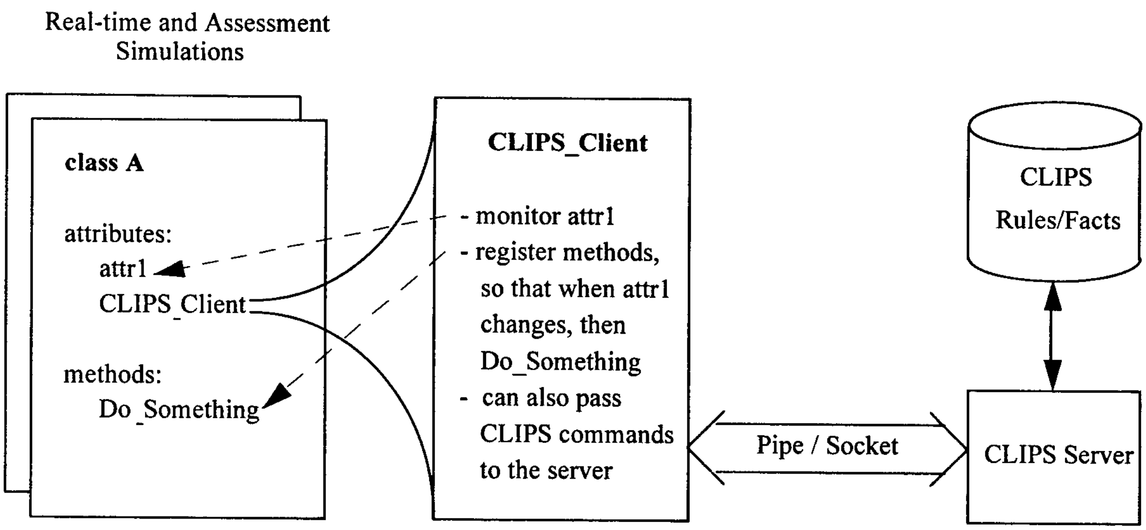


Figure 2. Expert System

The simulation is continually processed by the expert system. Data that reflects the current simulation state and all of its components is continually obtained and processed. As thresholds are breached, the expert system sends messages that specify the situations that have occurred to the real-time simulation. The information is then passed up to the controller and then to the display engine to alert the user of these situations so that appropriate actions can be taken.

3.7. Display Engine

The display engine is a separate process that drives the user interface to the Eagle View system. It caches a large space of state variables that are updated by the real-time and assessment simulations. The display engine presents both run-time animation of the real-time simulation and animation of the assessment simulation as an aerial view of the base. The base layout is drawn by using the IMDE map editor. Similarly, icons can be associated with the entities in the simulation by editing them in the map editor. The position of entities on the base is also defined using the map editor. Some entities, such as planes and trucks, are moving objects which update their location based on data from the data layer. The interface can be either mouse-driven or touch screen-based depending on the monitor's capabilities. Information views that provide detailed information can be displayed for entities in the simulation. The display engine also provides explosion features where the user can drill down into a particular building to obtain a detailed view of the interior.

4. Eagle View Prototype Scenario

The Eagle View application opens with a current status report of the base. The base is in normal operations mode, with a few missions being flown by aircraft in the squadron. Icons that represent buildings used to support the aircraft during regular operations and a deployment are seen in the application.

During the normal operations, all entities may or may not have a color attached to the icon. The lack of color indicates that the entity is functioning normally. When an entity turns red or yellow, something noteworthy has occurred that must be attended to by the wing operations commander. In the first case, a building turns yellow. The Eagle View user will click on the building, bringing up a window describing the characteristics of the chosen entity (Info View).

The Info View window details the building chosen. Apparently, the POL building is getting low on resources. The wing operations commander can then look at the current level of POL available, or view diagrams of historical data. The commander can also quickly assess the base for a specified number of days and view the results. This is done, and it appears that the resource level in POL will not be a problem for the foreseeable future. No action is taken.

In another scenario, the Wing Operations Center receives a message asking whether the commander could deploy 15 F16s for thirty days and be ready for the airlift in 48 hours. This question spawns many processes within the Eagle View application. First, a new plan must be created. The commander chooses "Deployment" under the Plans|Create menu and then selects what Unit Type Code (UTC) should be used and to what base the deployment is going. Eagle View then displays the UTC, showing a parts list and the quantities needed. At the touch of a button, Eagle View queries on-line systems to determine recommended quantities of each part based on current supplies in stock at the receiving base and other factors such as terrain and weather. The commander may then either accept or override Eagle View's recommendations. The commander then clicks another button and Eagle View creates a plan to implement the deployment.

After the plan is created, the commander assesses the plan to see if the deployment requirements can be met. The plan sends out notices to the supply buildings to generate palettes of the requested parts in the specified quantities, and selects the aircraft to be deployed. The first assessment shows that the base can meet the necessary requirements; the aircraft can be ready for a deployment in 48 hours. The commander then sends a message to the requesting officer replying that the wing can be ready for the airlift in 48 hours.

The Eagle View system then tracks the real-time status of the deployment preparations over the next 48 hours. However, as with all plans, not everything goes according to schedule and some problems do arise.

First, one of the 15 planes that were assigned for the deployment turns red on the screen. A click of this plane brings up the Info View for the aircraft; it says that a subsystem on the aircraft (the IFF transmitter) has failed and needs to be replaced. However, the commander does not have the time or the manpower to work on the aircraft. The commander then substitutes an idle aircraft for the deployment.

The next sign of trouble occurs when the scale used to weigh palettes as they arrive at the marshaling area has turned red. Again, the scale is clicked to bring up the Info View; the scale has broken down, but should be fully functional in about an hour. The commander says this is satisfactory; no action is needed at this time.

The commander receives another message requesting three additional aircraft (bringing the total to be deployed to 18). The commander examines the current situation, selects three aircraft that are currently idle for deployment, and answers the message in the affirmative.

Finally, the marshaling area turns red, signaling that something is wrong with the organization of the palettes into separate chalks for each cargo plane that will be arriving. A click on the area shows that one of the chalks is highlighted; a click on that item brings up an Info View that illustrates the problem. The configuration of the cargo planes that will be arriving has changed; a chalk that is already being built will have to be split into two to accommodate the change. The commander then goes into the plan files, opens the existing plans, and edits the current plan using the Plan Editor. The commander then enters the new configuration of chalks to match the arriving planes, and the change is implemented. The chalks construction will then conform to this new plan.

After these changes are made, more projections can be made throughout the 48 hour deployment period to ensure the deployment process is proceeding on schedule. With the Eagle View system, planning, observing, and changing the deployment process is quick and easy.

5. Future Directions

5.1. Real-time Simulation

The current real-time simulation operates primarily by simulating base activities. It processes a few simulated data feeds. In a production system, the real-time simulation would do very little simulation and would rely very heavily on actual real-time data. This type of approach would require a significantly different simulation than that of the prototype system.

5.2. Projection and Assessment Simulation

For purposes of the Eagle View prototype, the projection and assessment simulations execute a single run of the simulation. In order to gather statistically significant data from the projection and assessment simulations, it would be necessary to execute several runs and collate the data. The processes to gather the data from multiple runs and process it in a statistically valid manner require further investigation.

For purposes of the Eagle View prototype, the projection and assessment simulations use the same simulation as the real-time simulation. The projection/assessment simulations are started by using a fork() system call from the real-time simulation. This creates the projection/assessment simulation with values equivalent to the current state of the real-time simulation. As noted above, in a production system, the real-time simulation would be significantly different from this prototype. This would preclude the use of fork() to start the projection/assessment simulation from the current state of the real-time simulation. This means that a new method of initializing the projection/assessment simulation would need to be developed.

5.3. Simulated Data Injector

In the Eagle View prototype, real-time data feeds are only simulated. In a production system, this simulation would need to be replaced by a process which monitors incoming data from real time data feeds. As long as the real-time data can be translated to an event in the simulation, the data receivers developed for Eagle View can be extended to handle that data.

5.4. Display Engine

The display engine was created to be highly reusable and expandable. Simulations developed in IMDE can be designed to use the display engine. This would be accomplished by descending classes from model and then overriding various methods to provide information to the display engine. Much of the work to override the required methods could be automated in IMDE. This would allow a developer to "hook in" the display engine capabilities in new simulations with minimal effort.

6. Conclusion

Eagle View successfully demonstrates the capabilities and potential of an interactive wing level logistics management and planning tool. The display engine provides a real-time aerial view of base activities. Eagle View demonstrates the concept of taking real-world data to update the simulation through the use of the simulated data injector. Eagle View has been implemented with a foundation set of user interfaces, which can be reused or extended for use with other simulations. Furthermore, the layout of the base and the icons used to represent entities on the base may be modified using the IMDE Map Editor. Eagle View demonstrates situation assessment through the use of plans. The operator can assess a new or modified plan to allow the user to "look into the future" through simulation and see the possible consequences of implementing a plan. The operator can also project into the future based on the current plan. All of these features can be combined to result in a powerful tool that in the future may provide invaluable support to a wing commander and his or her staff.